Changil Kim | Teaching Statement

Teaching is my passion; it is one of the primary reasons that I want to become a professor. I spent my formative years as a researcher at ETH Zurich, Switzerland, which had a particular emphasis on education: every graduate student there teaches every semester. Thanks to this system, I had the opportunity to teach a number of courses at various levels and to take a differing role at each class, from giving a weekly recitation lecture for an introductory programming class for freshmen, to lecturing an advanced computer graphics class for graduate students, to organizing and mediating a seminar class based on presentations and discussions on cutting-edge academic papers. In addition, I have mentored students with varying degrees and backgrounds, as well as interns at the Disney Research laboratory which I was also affiliated with. Some of the people that I mentored are now at leading research universities, and others have started their own businesses (with one of them already being a millionaire).

Teaching experience. As a teaching assistant, each spring I gave weekly recitation lectures for one of the two foundational computer science classes: Informatik I, which would translate to CS 101, and Design of Digital Circuits. In particular, Informatik I was a large class with more than half a thousand students enrolled. In Switzerland, every student enjoys universal education and anyone who graduated from a high school ("gymnasium") may enroll at any university without taking an exam, although they have to pass a high bar to graduate. About 20 teaching assistants would meet on a weekly basis to share their progress, discuss exercises, plan next recitations, and so on. We were in charge of making exam questions, overseeing the exams, and grading them, as well as designing weekly exercises and planning materials for lectures. In sum, it was a comprehensive collaborative project for both the students and the teaching assistants.

In such a substantially sized class, any individual student could easily find themselves lost with no one to help them. To prevent this from happening, I particularly focused on the individual progress of the students in my recitation group. They were mostly freshmen (the rest being sophomores), with diverse backgrounds and a varying amount of exposure to computer science. Some had already developed their own smartphone apps in their free time, while some were learning to code for the first time. This experience taught me how to care about individual students with varying skill sets and differing progress paces. For those who were starting to code for the first time, I frequently used "whiteboard" coding, where I would, cooperatively with my students, solve a few questions in a top-down manner during recitation. That is, we would first write the rough skeleton of the code and then verbally work through the sub-algorithms needed, much like pseudo-code, which helped the students develop the ability to break a problem down into smaller pieces. Then we would turn each verbal instruction to a functional form, which we would expand into a full function definition later on, using each new component of coding they had just learned during the previous lecture. This was a very effective method to engage the students with the least programming experience and excite them with the fun, problem-solving nature of coding, so that they would not lose their interest in programming.

In contrast, each autumn I taught the graduate-level Computer Graphics class of about 30–50 students. From the very beginning of my teaching assistantship, I had been deeply involved in structuring the course with the lecturers (usually two professors). Two teaching assistants were responsible for running exercise sessions and designing programming and theoretical problem sets. Every year, we revised the exercises, often including more interactive elements, one of which was the use of WebGL. Students would implement various appearance models by writing shader code directly on the web interface. They could freely share their results with others and discuss them. Each exercise included several bonus questions to encourage motivated students to implement advanced features. Each week, we gave recitation lectures, graded exercises, and answered students' questions and suggestions. We introduced an interview-format evaluation session on the exercise, where students could show their solutions and justify their particular implementation or design choices in a more interactive manner. We also created all exam questions and graded them as well. Throughout the semester, students would build their own ray tracer as a semester project and participate in the rendering competition, which took place at the end of a semester.

As a postdoctoral researcher, I gave lectures for the Computer Graphics class at ETH Zurich and the graduate-level Advanced Computer Graphics class at MIT. I was responsible for topics like image-based rendering, stereoscopy, light fields, digital imaging, augmented and virtual reality, appearance modeling, and digital fabrication. In addition, I organized a seminar course for graduate students, Advanced Topics in Computer Graphics and Vision, where students presented and discussed influential academic papers. I curated the research papers and mediated discussions, in addition to recruiting mentors with relevant expertise and matching them to students while myself mentoring one.

Student mentoring. In addition to my formal teaching experiences, I mentored a number of undergraduate and graduate students (at both the masters and doctoral level) and interns at Disney Research. I felt fortunate to have the chance to guide someone so directly: as a mentor, I was able to understand my mentee's struggles and help them on their academic journey, while simultaneously learning myself how to best be an effective guide and teacher. This deeply motivated me to seek a faculty member as my career path. I had the unique experience of being affiliated with both an academic lab at a university and an industry lab. This allowed me to collaborate with the engineers and artists at Disney Studios and participate in a number of production projects, where I saw the techniques I developed being used for real-world projects. That enriched my understanding of computer graphics as both an academic and industrial discipline and gave me a broader perspective on how my research affects the world around me—a valuable perspective that I look forward to sharing with my own future students.

Proposed courses

I look forward to teaching the core undergraduate curriculum such as Discrete Mathematics, Algorithms and Data Structures, and Linear Algebra, as well as the following courses at undergraduate and graduate levels. Great emphasis will be placed on exercises and class projects so that students will gain practical knowledge and learn how to apply it to real-world problems.

Introduction to Computer Graphics. An undergraduate-level course, introducing students to 3D computer graphics. The course covers the fundamentals of 3D graphics, geometric modeling, photorealistic rendering, and animation. The basics of the relevant mathematical tools are also presented. In addition to practical and theoretical exercises, there are "artistic exercises," through which students experience the existing modeling and rendering tools to create content. A semester-long class project supplements the practicality of the learned material.

Mathematical Foundations for Computer Graphics and Vision. An undergraduate-level course. Fundamental mathematical tools needed to solve a variety of graphics and vision problems are presented. Linear algebra, numerical integration, optimization, finite element methods, machine learning techniques, high-performance computing, and GPU programming are addressed.

Computer Graphics Seminar. An advanced undergraduate or graduate-level course. Students are introduced to the recent research literature through weekly paper reading. It is emphasized to develop the ability to critically assess the state-of-the-art research works and participating in group discussions, much like an academic paper review process.

Deep Neural Networks for Visual Computing. A project-oriented advanced undergraduate or graduate-level course. This course addresses the various topics in machine learning with a strong focus on the recent advances in deep learning and its applications to computer graphics and vision. Beyond the fundamentals of deep neural networks, advanced topics such as generative models, image translation, metric learning, and multi-modal learning are covered.

3D Vision. A project-oriented graduate-level course. Advanced topics about 3D vision are covered: camera model and calibration, stereo vision and epipolar geometry, multi-view geometry, structure from motion, SLAM, shape from shading, optical flow, advanced sensors, and 3D shape recovery and surface reconstruction.

Advanced Image/Video Processing. A project-oriented graduate-level course. The fundamental theories about signal processing, optics, filters, image representation, image/video processing, Fourier analysis, and Bayesian inference are covered in depth, followed by advanced topics in statistical models, such as probabilistic graphical models, neural networks, and temporal models. Finally, the recent advances are addressed with cutting-edge research topics.

Computational Photography. A project-oriented graduate-level course. After a brief introduction to optics, the mechanisms of modern imaging devices are addressed. Various image reconstruction filters are discussed from the signal processing perspective. Advanced topics about light fields, coded imaging, computational cameras, and modern photodetectors are also covered.